YAHOO! search

Your Search: `C++ template parameter`   **Search**   Advanced Web Search
Preferences

| Web | Images | Directory | Yellow Pages | News | Products |

TOP 20 WEB RESULTS out of about 61,100

1. **Alexandre Oliva - Re: C++ template parameter** constraints
Re: C++ template parameter constraints. From ... Date: 30 Jan 2003 01:38:27 -
0200; Subject: Re: C++ template parameter constraints; Organization ...
gcc.gnu.org/ml/gcc/2003-01/msg01616.html cached | more results from this site

2. **C++ Template Parameter Constraints** (PDF)
1 ///-bs- Feb 19, 2001- , ICL, CS Dept., PKU from Newsgroup
comp.lang.c++.moderated C++ Template Parameter Constraints (Some newsgroup
posts collected between ...
icl.pku.edu.cn/bswen/css/C++%20Template%20Param%20Constraints.pdf view as
html | more results from this site

3. **Template Metaprograms (Todd Veldhuizen)**
... If your compiler does not yet support the ANSI C++ bool type, then an integer
template parameter works just as well. C++ version, Template metaprogram
version. ...
osl.iu.edu/~tveldhui/papers/Template-Metaprograms/meta-art.html cached | more
results from this site

4. **Constraints for Function Template Parameters in C++**
Constraints for Function Template Parameters in C++. C++ builtin template
parameter constraints. C++ provides a simple syntax for ...
m17n.org/martin/writings/template-parameter-constraints.html cached

5. **Comp.compilers: Re: C++ Template implementation**
... parameters was one of the stupider bits of syntactic design in C++. The way the
grammar is defined, the "first" possible '>' closes the template parameter list ...
compilers.iecc.com/comparch/article/98-04-093 cached | more results from this site

6. **Tech Talk about C++ Templates / Comeau C++ Template FAQ**
... Trying to use a template parameter , in a friend is ill-formed: template <typename
T ... code); According to Section 7.1.5.3 paragraph 2 of Standard C++: ...
www.comeaucomputing.com/techtalk/templates/ cached | more results from this site

7. **Larch/C++ Reference Manual - Template Specifications**
... s. The most common is a type-parameter , which usually ... type name in the body
of the template class or ... The C++ syntax uses class for declaring such formal
type ...
www.cs.iastate.edu/~leavens/larchc++manual/lcpp_153.html cached | more results
from this site

8. **ervan - c++/1209: Enums as template parameters produce "can't ...**
... When an integer-style template parameter is specified with ... enum GNetConn
{ GSimpleNetConn, GLongNetConn }; template <GNetConn T ... 0>', as given by
C++ RTTI and it ...
sources.redhat.com/ml/gdb-prs/2003-q2/msg00132.html cached | more results from
this site

---

9. **C++ Programming: templates as template parameters - supplying ...**
... keywords, limit to C++ Area, ... template for a parameter while supplying
constructor arguments to the template parameter, which is ...
www.experts-
exchange.com/Programming/Programming_Languages/Cplusplus/Q_20696885.html
cached | more results from this site

10. **[Bug c++/3671] cannot deduce enum template parameter with ...**
... [Bug c++/3671] cannot deduce enum template parameter with multiple
overloads. From: pinskia at physics dot uc dot edu; Subject: [Bug ...
www.mail-archive.com/gcc-bugs@gcc.gnu.org/msg38082.html cached | more results
from this site

11. **C++ Templates: Index**
... This is the index of all code examples of the book C++ Templates - The Complete
Guide by ... named template parameter nontype template nontype template
application. ...
www.josuttis.com/tmplbook/idx.html cached | more results from this site

12. **CUED: C++ templates**
... code swaps 2 integers in the usual C++ way. ... is potentially generic, templates
can be used - template <class T ... T" name is arbitrary (like a formal parameter in
a ...
www-h.eng.cam.ac.uk/help/tpl/languages/C++/templates.html cached | more results
from this site

13. **C++ Templates**
C++ Tips: Templates. ... templates without as return value or parameter ensuring
identical typedefs in templates restricting constant template arguments templ ...
cpptips.hyperformix.com/Templates.html cached | more results from this site

14. **C++: Templates** (PDF)
C++: Templates Miro Jurisic c meeroh@meeroh.org ... Function template
parameter deduction template <typename T> T max (const T& inLeft, const T&
inRight) { if ...
web.periodic-kingdom.org/People/Miro/Papers/MacTechGrp-
CppPresentations/Templates/Templates.pdf view as html

15. **Bookpool: C++ Templates: The Complete Guide**
... Lazy Instantiation. The C++ Instantiation Model. ... Functors as Template Nontype
Arguments. Function Pointer Encapsulation. ... Accessing Parameter Types. ...
www.bookpool.com/.x/37paghqu34/sm/0201734842 cached | more results from this
site

16. **C++ Template-Programmierung - CTE - DIGICOMP AG - Die ...**
... Einführung: C++-Wiederholung; Template-Grundlagen; Compilerunterstützung;
Einsatzbeispiele. ... Implementierung der Funktionen; Template-Parameter;
Template-Argumente; ...
www.digicomp.ch/kurse/CTE.html cached

17. **C++ Templates: The Complete Guide - Addison-Wesley and
Benjamin ...**
... Lazy Instantiation. The C++ Instantiation Model. ... Functors as Nontype Template
Arguments. Function Pointer Encapsulation. ... Accessing Parameter Types. ...
www.aw-bc.com/catalog/academic/product/0,4096,0201734842-TOC,00.html cached

---

| more results from this site

18. **Portability Hints: Borland C++ 5.5.1**
... they have significant problems with Microsoft Visual C++) and pass ... to std::string
or declare the template function as taking a const char * parameter. ...
www.boost.org/more/borland_cpp.html cached | more results from this site

19. **Proposed Addition to C++: Typedef Templates** (PDF)
... are not deducible when used as parameter types in ... template<typename T>
typedef T MyT; template<typename T ... page 7 Proposed Addition to C++: Typedef
Templates ...
www.gotw.ca/publications/N1406.pdf view as html | more results from this site

20. **Proposal to add template aliases to C++** (PDF)
... Email: mmarcus@emarcus.org Proposal to add template aliases to C++ 1. The ...
a simpler template is desired (say one requiring only a single template parameter).
...
anubis.dkuug.dk/jtc1/sc22/wg21/docs/papers/2003/n1449.pdf view as html | more
results from this site

Results Page:
1 2 3 4 5 6 7 8 9 10 ▶ **Next**

| Web | Images | Directory | Yellow Pages | News | Products NEW |

Your Search: `C++ template parameter`   **Search**   Advanced Web Search
Preferences

Search from anywhere online with the Yahoo! Companion Toolbar

Y!   Search Web   Search This Site   Highlight   Bookmarks

YAHOO! search

Your Search: [C++ template]   [Search]   **Advanced** Web Search
Preferences

[Web] [Images] [Directory] [Yellow Pages] [News] [Products]   powered by (hp)

Categories: • C++ Class Libraries > Active Template Library (ATL) • C++ Class Libraries
> Standard Template Library (STL)

SPONSOR RESULTS   (What's this?) (Become a Sponsor)

- **C++ Templates: The Complete Guide** C++ Templates: The Complete Guide.
  www.amazon.com
- **Get Self-Paced C+ Training for Beginner** Learn C+ Programming by certified trainers - Large selection, low prices, interactive CD-ROMs, videos and hands-on labs.
  www.youlearn.com

TOP 20 WEB RESULTS out of about 360,000

1. **Tech Talk about C++ Templates / Comeau C++ Template FAQ**
   Tech Talk About C++ Templates Comeau C++ Template FAQ. Copyright © 2000-2002 Comeau Computing. All rights reserved. THIS IS A VERY NEW PAGE UNDER CONSTRUCTION. ...
   www.comeaucomputing.com/techtalk/templates/ cached | more results from this site

2. **nklein software---products/geoma**
   C++ Template Classes for Geometric Algebras: ... As far as we know, this is the first publically available set of C++ template classes to implement them. ...
   www.nklein.com/products/geoma/ cached | more results from this site

3. **First C++ Template Programming Workshop (Proceedings)**
   Proceedings of the 2000. Workshop on C++ Template Programming. 10 October 2000, Erfurt, Germany. One of the most exciting research ...
   oonumerics.org/tmpw00/ cached | more results from this site

4. **Template Metaprograms (Todd Veldhuizen)**
   ... T. Veldhuizen, "Using C++ template metaprograms," C++ Report Vol. 7 No. 4 (May 1995), pp. 36-43. ... C++ version, Template metaprogram version. ...
   osl.iu.edu/~tveldhui/papers/Template-Metaprograms/meta-art.html cached | more results from this site

5. **freshmeat.net: Project details for Aapl C++ Template Library**
   ... Aapl C++ Template Library by Adrian Thurston - Saturday, February 2nd 2002 18:15 PDT, Section: Software. About: Aapl is a C++ template ...
   freshmeat.net/projects/aapl/ cached | more results from this site

6. **Matrix C++ template class library with full source code**
   Matrix template class library for performing matrix algebra calculations in C++ programs for engineering / scientific works in easy and efficient manner. ...
   www.techsoftpl.com/matrix/ cached | more results from this site

7. **The Mighty C++ Template**
   ... The Mighty C++ Template. Can't find the information you're after? Try searching

---

our forums instead! By: Mitchell Harper Published ...
www.devarticles.com/art/1/8 cached | more results from this site

8. **Boost**
   provides free peer reviewed portable C++ source libraries.
   www.boost.org/ cached | more results from this site
   ↳ More sites about: C++ > Class Libraries

9. **Florian Schintke - Re: The future C++ template model in gcc**
   Re: The future C++ template model in gcc. To: Gabriel Dos Reis <gdr at codesourcery dot com>; Subject: Re: The future C++ template model in gcc; ...
   gcc.gnu.org/ml/gcc/2001-07/msg01906.html cached | more results from this site

10. **Linear Algebra with C++ Template Metaprograms**
    ... Siek. Linear Algebra with C++ Template Metaprograms. Rapid linear algebra is just one use. Todd Veldhuizen and Kumaraswamy Ponnambalam. ...
    www.ddj.com/documents/s=1722/ddj9608d/ cached | more results from this site

11. **C++ Template-Programmierung - CTE - DIGICOMP AG - Die ...**
    C++ Template-Programmierung (CTE). ... Kursinhalt. Einführung: C++-Wiederholung; Template-Grundlagen; Compilerunterstützung; Einsatzbeispiele. ...
    www.digicomp.ch/kurse/CTE.html cached | more results from this site

12. **Net.Objectdays 2000 -- Konferenz: C++ Template Workshop**
    Net.ObjectDays : Konferenz : C++ Template Workshop. Home. Aktuelles. Termine. Tagungsort. ... Kontakt. Suche. Sitemap. Site info. Workshop über C++ Template Programmierung. ...
    www.netobjectdays.org/node00/de/Conf/tmpw.html cached | more results from this site

13. **The Code Project - C++ Template class to walk through a tree - ...**
    All Topics, MFC / C++ >> C++ / MFC >> General C++ Template class to walk through a tree By Nandagopal An article decribing the use of templates in writing a ...
    www.codeproject.com/cpp/walkerarticle.asp cached | more results from this site

14. **PGI Workstation User's Guide - 12 C++ Template Instantiation**
    << >> Title Contents Index Home Help 12 C++ Template Instantiation. ... C++ also dictates that unreferenced template functions should not be compiled. ...
    www.pgroup.com/ppro_docs/pgiws_ug/pgiug_13.htm cached | more results from this site

15. **C++ Template Programming - The Template Seminar**
    The most comprehensive seminar on C++ template programming. ... Keep up with the C++ community. Understand generic programming and template metaprogramming. ...
    www.langer.camelot.de/Courses/Templates.htm cached | more results from this site

16. **SGI - Services & Support: Standard Template Library Programmer's ...**
    A freely available implementation of the C++ Standard Template Library, including hypertext documentation. SGI Logo, How to Buy Resellers ...
    www.sgi.com/tech/stl/ cached | more results from this site

17. **Janus - a C++ Template Library for Parallel Dynamic Mesh ...**
    ... We propose Janus -- a C++ template library of container classes and communication primitives for parallel dynamic mesh applications. ...
    citeseer.nj.nec.com/gerlach98janus.html cached | more results from this site

---

18. **Sun C++ template closure patch**
    ... Sun C++ template closure patch. To: Mumit Khan <khan@NanoTech.Wisc.EDU>; Subject: Sun C++ template closure patch; From: "John W. Eaton" <jwe@bevo.che.wisc.edu>; ...
    www.octave.org/octave-lists/archive/octave-maintainers.2001/msg00030.html cached | more results from this site

19. **c++ template values in gdb**
    ... c++ template values in gdb. From: Srirang K. Karandikar. Subject: c++ template values in gdb. Date: 21 Nov 2000 17:39:32 GMT. ...
    mail.gnu.org/archive/html/bug-gdb/2000-11/msg00032.html cached | more results from this site

20. **chastain - gdb/40: C++ template functions have return types in ...**
    gdb/40: C++ template functions have return types in their names. To: gdb-gnats at sources dot redhat dot com; Subject: gdb/40: C++ ...
    sources.redhat.com/ml/gdb-prs/2001-q1/msg00044.html cached | more results from this site

Results Page:
1 2 3 4 5 6 7 8 9 10 ► **Next**

[Web] [Images] [Directory] [Yellow Pages] [News] [Products] **NEW!**

Your Search: [C++ template]   [Search]   **Advanced** Web Search
Preferences

Save time with the Yahoo! Search Toolbar

[Y!] [_____] [Search Web] [Search This Site] [Highlight] [Bookmarks]

YAHOO! search

Your Search: "interface template parameter"   [Search]   Advanced Web Search
Preferences

| Web | Images | Directory | Yellow Pages | News | Products | |

TOP 2 WEB RESULTS out of about 2

1. Object Interconnections 1 Introduction 2 The Socket Client ... (PDF)
    ... conformant interface. Template parameter- ization is a useful technique that
    increases the flexibility and portability of the code. ...
    www.iona.com/hyplan/vinoski/col3.pdf view as html

2. DD-Designer (PDF)
    Page 1. 1 DD-Designer Dual Dynamics Editor and Code Generators Tool Version:
    May 18, 2000 10:08 pm Docu Status: REVIEW (please report ...
    www.ais.fraunhofer.de/INDY/herbert/PraktikumWinter01/DD-Designer.pdf view as
    html

| Web | Images | Directory | Yellow Pages | News | Products NEW |

Your Search: "interface template parameter"   [Search]   Advanced Web Search
Preferences

The Mighty C++ Template

Author: Mitchell Harper
Date Added: 23rd Nov 2001
Type: Tutorial
Rating: ■■■■■■■■■□

This is a printable version of "The Mighty C++ Template". For the complete online version, please visit
http://www.devarticles.com/content.php?articleId=8

Page 1: Introduction

One of the biggest advantages of using C++ is templates. Templates were designed from the ground up
to allow developers to write one function to handle many different types of parameters. Because C++ is
a strongly typed language (ie: You must declare a variable before you can reference it), each and every
function you create must specify the data type of each parameter it accepts and also the data type of
its return value, such as int, bool, char, string, etc.

Page 2: Understanding function overloading

To understand where templates have come from though, you must first understand function
overloading. Let's say for an example, that we wanted to create a function that would work with
numeric values and return the average of those values. The function should be able to accept integers,
floats and doubles. To accomplish this, we could create three functions to handle each different type of
numeric value:

To work with integers:

```
int GetAverage(int num1, int num2, int num3)

{

return static_cast<int>((num1 + num2 + num3) / 3);

}
```

To work with floating point values:

```
float GetAverage(float num1, float num2, float num3)

{

return static_cast<float>((num1 + num2 + num3) / 3);

}
```

To work with double values:

```
double GetAverage(double num1, double num2, double num3)

{

return static_cast<double>((num1 + num2 + num3) / 3);

}
```

Then, whenever we called the GetAverage() function from our program, the C++ compiler would decide which function it should call based on the types of the parameters passed to that function. If an un-handled type, such as long was passed to the function, then the compiler would automatically cast it to the type that it most closely resembled (In this case, long would be converted to int).

As you can see, we end up with three functions that do exactly the same thing. This is bad, because it creates a lot of redundant code, and our executable file will be bigger than it needs to be.

There has to be a better way... and there is... it's called a template.

Page 3: The template

If you've ever programmed with Visual Basic, then you will probably be familiar with the variant data type. The variant is type-less in a sense that it allows a variable to be declared as a wild card and contain any type of data.

Well, C++ doesn't allow this, and in many ways, templates can substitute for variants. To fully understand templates, let's start with an example. Open your favourite C++ IDE, and create a new project with two files: main.h and main.cpp. Into main.h, enter the following code:

```
template<class T> T GetAverage(T num1, T num2, T num3);
```

This is our template declaration. It looks like a regular function declaration and its purpose is exactly the same as any functions declaration: to let the compiler know that the template exists. Let's break down the templates declaration:

```
template<class T>
```

This section of the declaration tells the C++ compiler that we are defining a new template definition. Between the angled brackets, we are declaring a new type identifier, T. This tells the compiler to create a new type identifier called T. T will hold the type of variable that this template will be working with. The type of variable is decided from the types of each of the parameters passed to the function. If, for example num1, num2 and num3 were of type int, then T would be an int.

```
T GetAverage(T num1, T num2, T num3);
```

Let's step away from templates for a moment and imagine the following function declaration:

```
int GetAverage(int num1, int num2, int num3);
```

Looks similar to our template definition right? Well it is. The T is acting as the type of variable that the template will work with. The template doesn't know what type of variable it can accept yet. As long as the variables passed to the function can handle the operator+ and operator/ (can be added with/divided by other variables) functions (which all numeric types can), then the function will work successfully and will return the sum of all of the numbers divided by three (the average).

Page 4: The template (contd.)

Now that we have created the function declaration for our template, lets create the actual code for it. In main.cpp, enter the following code:

```
#include <iostream>

#include "main.h"

using namespace std;

template<class T> T GetAverage(T num1, T num2, T num3)

{

return static_cast<T>((num1 + num2 + num3) / 3);

}

int main()

{

int x = 10;

int y = 20;

int z = 30;

int returnValue = GetAverage(x, y, z);

cout << "Returned " << returnValue << endl << endl;

return 0;

}
```

We start out by #including the iostream head filer (so we can output to the screen) and our main.h header file, which contains the declaration of our template.

Next, we have the full code for our template:

```
template<class T> T GetAverage(T num1, T num2, T num3)
```

```
{

return static_cast<T>((num1 + num2 + num3) / 3);

}
```

The first line is exactly the same as the templates declaration in main.h. Our template only contains one line, which calculates the average of the three variables passed in as parameters, and returns that value.

Our template uses the static_cast<>() function to make sure that the return value is also of type T. This is necessary, because most of the time, when finding the average, the result will contain a decimal point and mantissa. If, for example, we passed three int values to the template, then we would expect an int to be returned, and not a double, for example. That's what the static_cast<>() function is responsible for handling.

Note: The static_cast<>() function is actually a template. The value between the angled brackets tells the template what the return type of the template should be.

Before moving onto the next section, try changing the types and values of x, y, and z. The return type of the template will be the same as the values passed to it!

Page 5: A simple class template

Templates can also be used to create classes. Class templates stem from the same principles as normal templates, and are used in exactly the same way. A class template allows a developer to create classes whose constructor, destructor and functions can accept different types of values, just like a function template.

To create a class template, create a new C++ project and add two files: classtemplate.h and classtemplate.cpp. Into classtemplate.h, enter the following code:

```
template<typename T> class MyClass

{

public:

MyClass(T x = 0, T y = 0, T z = 0) : n1(x), n2(y), n3(z) {}

T GetAverage();

private:

T n1;

T n2;
```

```
T n3;

};
```

This code might look a little confusing at first, but allow me to explain it. Notice how we have defined the entire class in the header file this time, and not just the declaration like we did for the function template?

Firstly, we have the class declaration:

```
template<typename T> class MyClass
```

A class template declaration is a bit different to a function template declaration. Firstly, between the angled brackets, we are creating a type identifier. We have used "typename" instead of "class" here, but they are interchangeable.

Next, we have the "class" keyword followed by the name of the class, "MyClass". The class can be called anything you like, but we will use "MyClass" in this example.

```
{

public:

MyClass(T x = 0, T y = 0, T z = 0) : n1(x), n2(y), n3(z) {}

T GetAverage();

private:

T n1;

T n2;

T n3;

};
```

Following the template class's declaration, we have its default constructor and one function named GetAverage() which takes no parameters and returns a value of type T.

Notice that our constructor is accepting three variables of type T, all with a default value of zero. The constructor assigns these values to our private member variables (of type T) n1, n2 and n3 respectively. These private variables are simply used to hold the values of the variables passed in and can't be accessed.

Page 6: A simple class template (contd.)

Open classtemplate.cpp and enter the following code:

```
#include <iostream>

#include "classtemplate.h"

using namespace std;

template<typename T> T MyClass<T>::GetAverage()

{

return static_cast<T>((n1 + n2 + n3) / 3);

}

int main()

{

MyClass<int> mc(1, 2, 3);

cout << "Return value is: " << mc.GetAverage() << endl << endl;

return 0;

}
```

Lets start by describing the GetAverage function of the MyClass class.

```
template<typename T> T MyClass<T>::GetAverage()

{

return static_cast<T>((n1 + n2 + n3) / 3);

}
```

Firstly, our class function is declared as a template using the template<typename T> keywords, meaning that our class function has just one type identifier, T.

```
T MyClass<T>::GetAverage()
```

Secondly, our class function returns a value of type T. Notice that the class name, "MyClass", is followed by our type parameter, T. This tells the C++ compiler that our function will handle values of type T.

The code for our class function is similar to the code for our template function, from above:

```
return static_cast<T>((n1 + n2 + n3) / 3);
```

All of the three privately declared variables, n1, n2 and n3 are summed up, divided by three and then returned as a variable of type T.

Page 7: Instantiating the class

Before we can use our class template, we must create an instance of it within our main function:

```
MyClass<int> mc(1, 2, 3);
```

```
cout << "Return value is: " << mc.GetAverage() << endl << endl;
```

On the first line, we are creating a new instantiation of our class template. Between the angled brackets, we are telling the C++ compiler that our class will be handling integer (int) type variables. This is a mandatory inclusion and the compiler will complain if you leave it out. The new instantiation of our "MyClass" template is called "mc". We are passing three values to the default constructor, which will be stored as the private variables n1, n2 and n3. In our examples, these variables will be of type int, meaning T will take on the role of an integer.

On the second line above, we simply call the GetAverage() function of our class template, just like we would with any class, outputting the value returned from the function, which looks something like this:

Return value is: 2

Page 8: Behind the scenes

You might be wondering just exactly what goes on under the hood when a C++ class template is instantiated.

Firstly, the C++ compiler creates an instance of the template and stores it on the stack. Because our class template passes integer values to its constructor, the compiler creates an instance of our class (on the stack) that can work with variables of type int. This leads us to an important point: The class template itself doesn't actually do any of the code processing. It simply creates a new instance of the class and its encapsulated functions for the type of variable that it is dealing with, such as int, float, char, string, etc. If an instance of a class for a specific variable type has already been created, the compiler will use that class to create the new instance. When the program ends, the class template and all of its copies are popped off the stack. Just remember that the template is only responsible for creating an instantiation of the class for the type of variable that is required by the program code and doesn't actually get executed.

Page 9: Conclusion

Templates provide the C++ developer with a whole new level of power, flexibility and robustness. Templates play a big role in several C++ libraries including the standard template library (STL) and the active template library (ATL), and fully understanding and utilizing templates where they are needed can increase the speed, cleanliness and re-useability of your code.

We have just briefly touched on templates in this article. If you want to know more about templates, take a look at the links and books shown below.

For more great programming articles, please visit http://www.devarticles.com. If you have an opinion or question about this article, then please post it at the devArticles developer forum, which is located at http://www.devarticles.com/forum